Remaillage parallèle pour le calcul haute performance

Arnaud Bardoux

Master 2 de Science, mention Informatique,
spécialité Réseaux Informatiques et Systèmes Embarqués

Présenté par

Arnaud Bardoux
arnaud.bardoux@etu.unistra.fr

Remaillage parallèle pour le calcul haute performance

Encadré par

Cédric Lachat
cedric.lachat@inria.fr

Au sein de

INRIA Bordeaux - Sud-Ouest
Équipe TADaaM

En collaboration avec

Christophe Geuzaine
Université de Liège

Jean-François Remacle
Université catholique de Louvain

Remerciements

Je tiens à remercier tout particulièrement Cédric Lachat pour m'avoir proposé ce projet qui m'a permis de développer mes compétences. Merci du temps que tu m'a consacré pour me permettre d'acquérir d'acquérir des méthodologies de travail. Ta bonne-humeur et ton accueille chaleureux ont permis à ces quelques mois d'êtres conviviaux.

Je remercie également Christophe Geuzaine et Jean-François Remacle pour l'aide qu'ils m'ont apporté pour la réussite de ce projet.

Sommaire

Chapitre 1

Introduction

1.1 Présentation générale

1.1.1 Inria - Institut National de Recherche en Informatique et en Automatisme

Créé en 1967 avec le site historique de Paris Rocquencourt, Inria est un institut national de recherche, dont le but est d'éduquer le pays dans les sciences de l'informatique et de l'automatique. Pour cela deux objectifs ont été définis. Le premier est de communiquer les résultats des recherches, par l'intermédiaire de publications, de participations et d'organisations de grands colloques, ceci afin de permettre un rayonnement international des nouvelles technologies issues de l'innovation française. Le second objectif est de transférer les connaissances et les technologies vers l'industrie au moyen de brevets, licences, partenariats, et de créations d'entreprises. Ainsi, Inria possède plus de 300 brevets actifs, et a permis la création d'une centaine d'entreprises.

Afin de définir les objectifs de recherches, des plans quadriennaux sont mis en places. Actuellement les objectifs de recherches sont délimités suivant cinq thématiques, qui sont :

- Mathématiques appliquées, calcul et simulation ;
- Algorithmique, programmation, logiciels et structures ;
- Réseaux, systèmes et services, calcul distribué ;
- Perception, cognition, interaction ;
- Santé, biologie et planète numérique.

Inria est structuré selon un modèle de projet-équipe (PE). Ce sont des regroupements de chercheurs centrés sur des objectifs scientifiques d'une thématique approuvée par l'institut. Les PE sont créés pour 4 ans, avec l'approbation d'une commission qui étudie l'intérêt du projet de recherche. Cette durée peut être renouvelée deux fois, ce qui porte la durée de vie maximale des PE à 12 ans. Cependant, la durée moyenne est de 8 ans. Ces cycles courts permettent à Inria de toujours être centré sur des sujets de pointe. De plus, les PE possèdent une grande autonomie que se soit sur l'organisation, ou sur la gestion.

Les PE sont constitués d'une vingtaine de scientifiques. Ils peuvent être exclusivement des chercheurs Inria, ou des membres d'établissements partenaires (universités, centres de recherches...). De plus, l'esprit de partenariat est très développé, que se soit avec l'industrie ou au sein même d'Inria. En effet, les transferts inter PE de connaissances, de ressources humaines, etc sont encouragés.

Afin de renforcer l'esprit collaboratif des PE, Inria a mis en place des Inria Labs, qui permettent de collaborer avec d'autres partenaires, académiques ou industriels. Leur objectif est de se regrouper autour de projets d'envergures nécessitant des compétences variées. Il existe quatre types d'Inria Labs.

Inria Project Lab regroupement de plusieurs PE, et éventuellement d'autres structures académiques. Leur but est, à travers un projet de recherche commun, de se concentrer sur une question scientifique d'actualité ;

Inria Joint Lab collaboration forte avec un partenaire privé sur un problème issue de la R&D de l'entreprise ;

Inria Innovation Lab association d'une PE avec une PME afin de renforcer la capacité d'innovation de la PME ;

Inria International Lab coopération entre des équipes Inria et des partenaires académiques étrangers afin d'organiser la présence d'Inria dans différentes régions du monde.

1.1.2 Équipe TADaaM - Topology-Aware system-Scale Data Management for high-performance computing

Dirigée par Emmanuel Jeannot, l'équipe TADaaM est composée d'une vingtaine de membres réunis autour de trois problématiques qui sont :

— *abstraire et modéliser la topologie des ressources et le comportement de l'application* ;
— *définir une bibliothèque propre et claire capable de passer à l'echelle* ;
— *créer un ensemble de mécanismes permettant d'optimiser l'exécution des applications.*

Elle travaille à créer une couche intermédiaire rassemblant toutes les données relatives à l'exécution des applications afin de prendre les meilleures décisions pour optimiser l'exécution des applications. Ces décisions peuvent concerner le placement des processus sur un cluster, comme l'ordonnancement des tâches... Cette couche intermédiaire prend la forme de bibliothèques abstrayant le matériel, la pile logiciels et les appels systèmes. Elles sont conçues pour des applications hybrides utilisant différentes technologies de parallélisme, comme MPI[1], OpenMP[2], les BLAS[3] et Pthreads[4]. Ceci permet d'avoir aussi bien une optimisation indépendante pour chaque application pour l'ordonnancement des threads... que des optimisations communes à toutes les applications qui permettent d'avoir une meilleur utilisation du réseau, du système de stockage...

Afin que cette couche intermédiaire puisse prendre les décisions optimales, elle doit connaître le comportement des applications quelle doit administrer. Les applications doivent donc définir leurs besoins en ressources et décrire leurs comportements. Pour cela, les bibliothèques mettent à disposition des API de haut niveau, permettant ainsi de répondre à deux caractéristiques clefs :

— Indépendance avec l'application et le modèle de programmation. Les développeurs n'ont donc pas besoin de réécrire l'application, juste ajouter les éléments de sémantique de haut niveau ;
— La facilité d'expression des besoins en utilisation des ressources et en interactions avec l'environnement. Connaissant bien leurs applications les développeurs peuvent le faire sans difficultés.

Étant communes à l'ensemble de l'environnement parallèle, les optimisations se font à l'exécution, permettant une plus grande flexibilité que des optimisations statiques, et donc une meilleur utilisation des ressources. De plus, elles ne modifient que très peu l'exécution des applications tout en gérant plusieurs parties de l'environnement. Enfin, comme cette couche intermédiaire est transversale, elle prend en compte les besoins de plusieurs applications en même temps, et gère leurs interactions.

L'équipe TADaaM travaille actuellement sur cinq projets de recherches qui ont abouti à des programmes. Ils sont les suivants :

Hardware Locality (HWLOC) Sert à découvrir et exposer la topologie des machines (processeurs, cœurs,...), afin d'aider l'application à adapter son comportement en fonction du matériel. Cela va de l'ordonnancement de threads au placement de processus MPI ;

New Madeleine Bibliothèque de communications inter processus visant à balayer l'ensemble des techniques d'optimisation de communication ;

TreeMatch Bibliothèque qui optimise le placement des processus en se basant sur la topologie du cluster et sur le schéma de communication de l'application ;

Scotch Bibliothèque parallèle et séquentielle pour le tri de matrices peu denses, et pour le placement et le partitionnement de graphes, d'hypergraphes et de maillages.

PaMPA Bibliothèque de gestion de maillages distribués permettant le remaillage parallèle.

Le domaine d'application choisi par l'équipe pour l'ensemble des ces projets est la gestion des maillages. Ceux-ci amènent des challenges importants, que se soit en terme de performances, de localité, de passage

1. MPI (Messages Passing Interface) : Bibliothèque d'envois de messages pour systèmes distribués
2. OpenMP : Environnement de parallélisation multithreads
3. BLAS (Basic Linear Algebra Subprograms) : Bibliothèque de manipulation de matrices
4. Pthreads : Bibliothèque POSIX de manipulation de threads

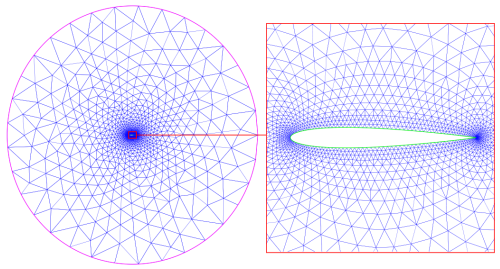
à l'échelle ou de gestion des données. En effet, ceux-ci sont composés de plusieurs millions, voir milliards, d'éléments.

1.1.3 Sujet d'étude

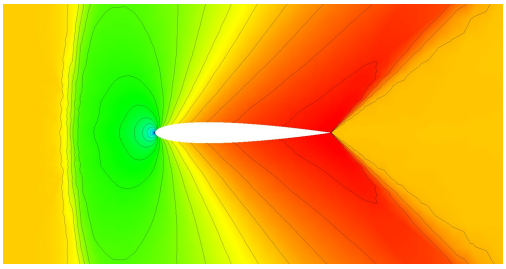
Tous les domaines se rapportant à l'étude d'un phénomène physique ont recours à la simulation numérique, que ce soit dans l'industrie : l'aérospatial, l'automobile..., ou la recherche : la physique nucléaire, la météorologie... Les simulations numériques sont utilisées soit pour des raisons de coût ("crash test" automobile), ou de faisabilité (étude de la propagation d'ondes). Ces simulations sont l'analyse de phénomènes physiques dans une représentation discrète du milieu d'étude, qui peuvent être en deux dimensions (étude des côtes marines) ou en trois dimensions (visualisation de l'écoulement de l'air autour d'un avion). Cette représentation, que l'on appelle maillage, permet de calculer des valeurs physiques en des points précis du domaine étudié afin d'en calculer l'évolution temporelle, qui est la solution attendue. La précision de la solution, donc de la simulation, est en corrélation directe avec la qualité et la finesse du maillage.

Prenons comme exemple la figure 1.1. La figure 1.1(a) montre le maillage de l'air entourant une aile d'avion. La figure 1.1(b) est la solution obtenue à partir de ce maillage. On peut constater que les isolignes ne sont pas nettes. Ceci est dû au fait que le maillage n'est pas assez fin à ces endroits. Donc, pour avoir plus de précision sur la solution, deux possibilités s'offrent à nous. Soit nous faisons un maillage entièrement fin, qui nous donnerait une solution très précise en tous points. Mais nous n'avons pas besoin d'une précision aussi importante sur l'ensemble du volume (ou de la surface). Cette solution gaspillerait donc du temps de calcul.

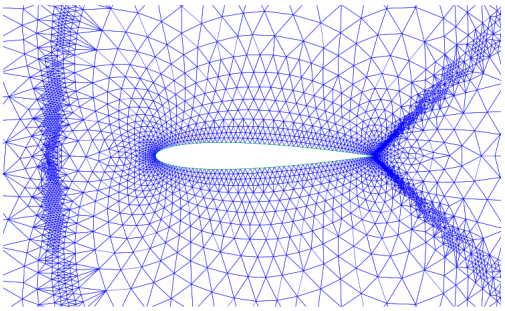
La deuxième option, illustrée par la figure 1.1(c), est de ne modifier que les zones correspondant aux besoins de précisions. La solution issue de la simulation sur ce maillage (figure 1.1(d)) permet de voir plus nettement les isolignes. Cette exemple nous permet de voir la corrélation directe entre la qualité des maillages et la précision obtenue après la simulation.



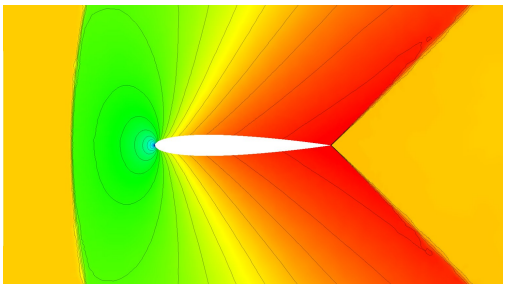
(a) Maillage initial



(b) Solution initiale



(c) Maillage après remaillage



(d) Solution après remaillage

FIGURE 1.1 – Solutions en fonction de la qualité du maillage

Comme les besoins en précisions sont en perpétuel augmentation, la taille des maillages ne fait que croître, si bien qu'aujourd'hui nous dépassons le milliard d'éléments. Le temps nécessaire pour générer ou remailler

de tels maillages sur un poste de travail est beaucoup trop important pour être réaliste. De plus, la quantité d'informations à stocker pour remailler des maillages aussi imposants dépasse largement la capacité mémoire des ordinateurs actuels. Pour ces raisons, il devient nécessaire de paralléliser le remaillage afin de rendre le temps d'exécution acceptable, ceci en répartissant les maillages à travers un réseau d'ordinateurs.

1.1.4 Définitions utiles

Afin de lever toute ambiguïté, nous avons besoin de définir des notions importantes. Les premières concernent les maillages et le remaillage.

Définition 1 (Maillage - représentation discrète) *Un maillage est une représentation discrète d'un milieu physique. Il permet, en des points précis du milieu, de calculer des valeurs suivant des contraintes définies par l'utilisateur.*

Définition 2 (Algorithmes de maillage) *Par maillage nous pouvons aussi faire référence aux algorithmes permettant d'obtenir les représentations discrètes que l'on appelle maillage. Le plus connue est la triangulation de Delaunay [1].*

Définition 3 (Remaillage) *Au cours d'une simulation les besoins en précision peuvent changer en certains endroits. Il est donc nécessaire de modifier le maillage en conséquence. C'est pourquoi il existe des algorithmes de remaillages.*

Enfin, la dernière notion dont nous allons avoir besoin, est la notion de systèmes distribués.

Définition 4 (Systèmes distribués) *Un système distribué est un ensemble de ressources interconnectés par un réseau de communication rapide. Ici, nous appellerons systèmes distribués un ensemble d'ordinateurs ayant leur propre mémoire, et communicant à travers un réseau. Nous pourrons utiliser les termes clusters ou systèmes à mémoire distribuée pour nous y référer.*

Ceci nous permet de donner la définition d'un nœud de calcul.

Définition 5 (Nœud de calcul) *Un nœud de calcul est une unité du système distribué. Ici, c'est l'un des ordinateurs composant le cluster.*

1.2 Contexte

1.2.1 PaMPA au sein de l'Inria

Afin de faire face à ce besoin grandissant en puissance de calcul, différentes méthodes de calcul de remaillage sur systèmes distribués ont été étudiées. La première qui vient à l'esprit est de paralléliser les opérateurs[5] effectuées par les remailleurs. Elle n'est que très peu utilisée, car elle est difficile à mettre en œuvre, et le nombre de communications, engendré par la distribution du maillage, augmente de façon importante à mesure que le nombre nœuds de calculs augmente. La seconde méthode est basée sur un processus itératif, et consiste à extraire des zones indépendantes les unes des autres, depuis un maillage distribué, afin de les faire remailler par un remailleur séquentiel sur différents nœuds de calculs.

Principe de PaMPA

François Pellegrini et Cécile Dobrzynski ont donc proposé à Cédric Lachat une thèse dans le but d'approfondir cette méthode. Ils ont créé la bibliothèque PaMPA [2] (pour PArallel Mesh Partitioning and Adaptation). Grâce à des algorithmes de partitionnement de graphe PaMPA peut remailler un maillage par zones en suivant un processus itératif en cinq étapes, illustré par la figure 1.2.

La première étape consiste à marquer les éléments de maillage (1) ne correspondant pas aux critères du numéricien. Dans le graphe des éléments à remailler, des zones de charges de calcul de remaillage équivalentes sont identifiées (2) pour ensuite être extraites (3) et réparties sur différents nœuds de calculs pour répartir

5. Opérateurs de remaillage : insertion, suppression, bougé de points, retournement d'arêtes

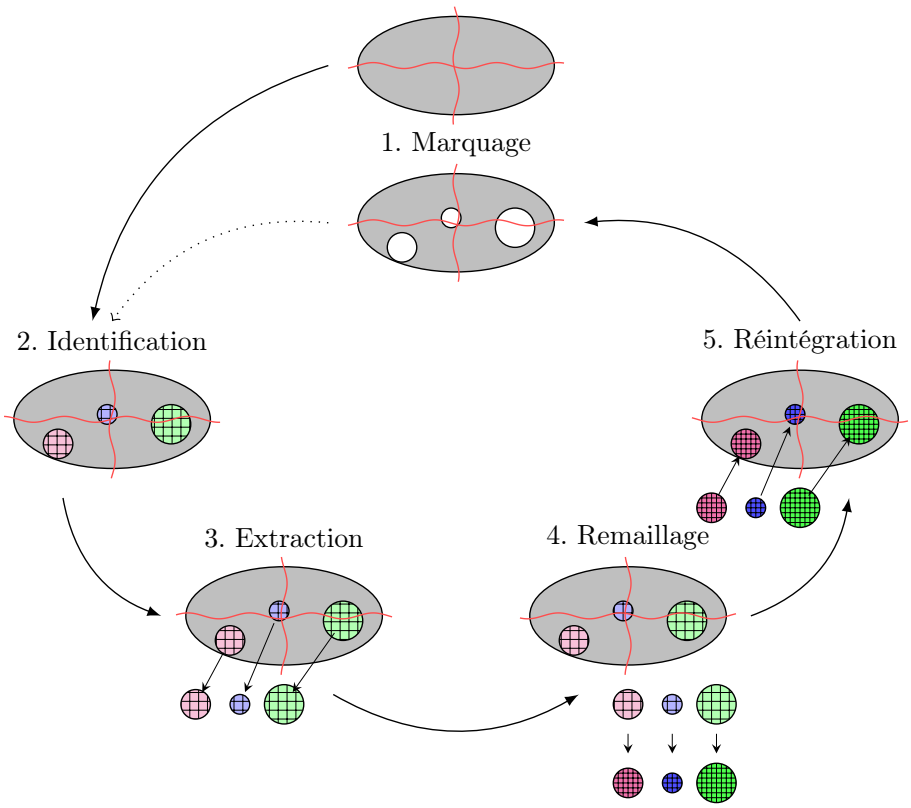


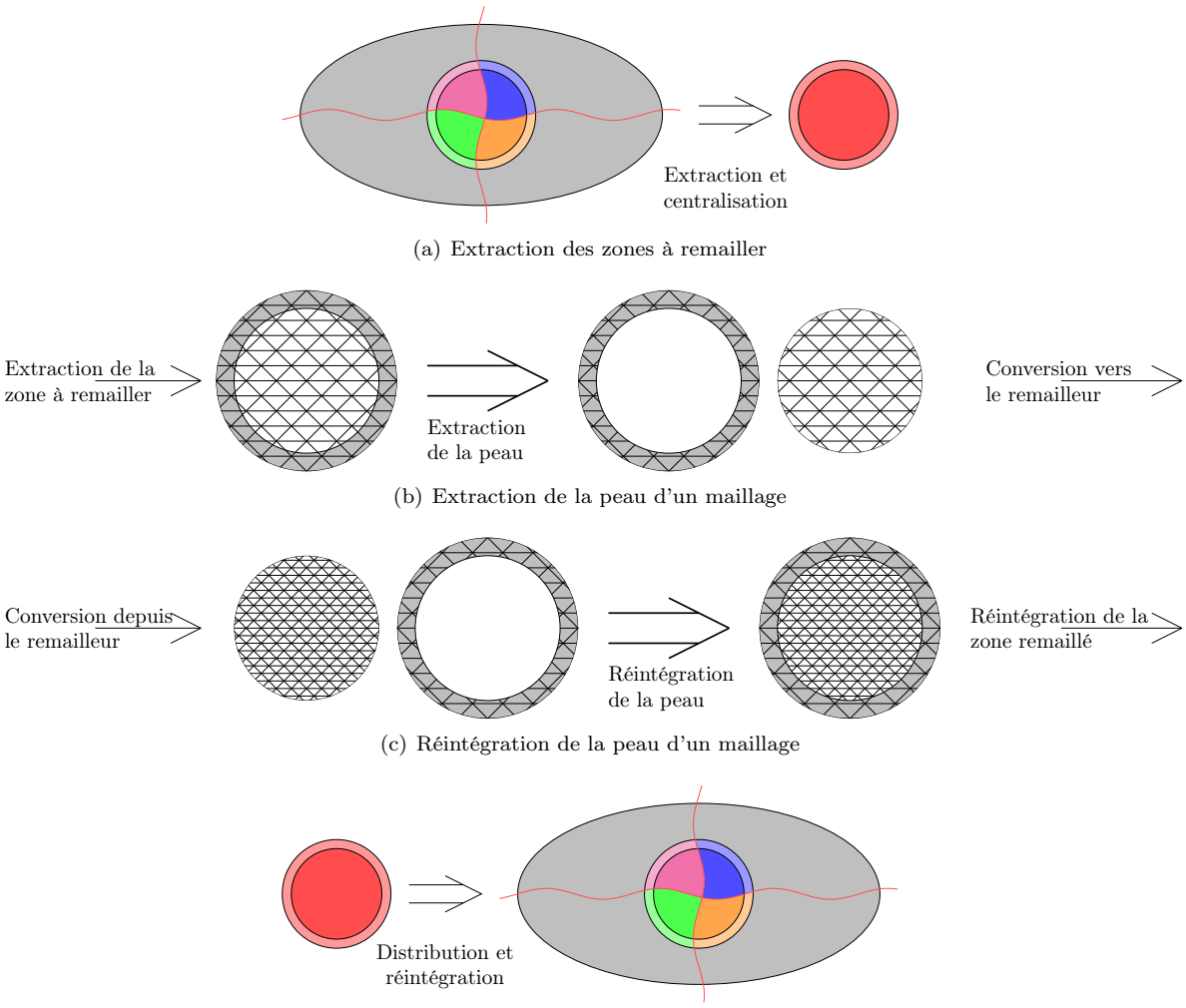
FIGURE 1.2 – Processus itératif en cinq étapes

la charge de manière équitable tout en minimisant les communications lors de l'extraction. Chacune de ces zones est considérée comme un maillage centralisé à part entière. Elles sont alors remaillées séquentiellement (4) sur des unités de traitement différentes. Enfin, ces maillages sont redistribués et réintégrés (5) au maillage distribué global. Ces maillages peuvent être réintégrés car la zone frontière entre ces parties a été protégée pour qu'elle ne soit pas remaillée. Ces cinq étapes sont répétées tant qu'il y a des éléments marqués à remailler, ou qu'en sortie de deux itérations successives les proportions d'éléments à remailler sont identiques. En effet, cela signifie que le remailleur n'est pas en mesure de modifier les éléments ne respectant pas les critères de qualité de l'utilisateur.

La bibliothèque PaMPA a été initialement liée au remailleur Mmg3d, et afin de tester sa généricité, nous lui intégrons les remailleurs TetGen et Gmsh. Pour ceci nous devons comprendre les actions faites des étapes 3 à 5. Nous allons donc les détailler en nous appuyant sur la figure 1.3.

Pour l'étape 3 qui extrait les zones à remailler (figure 1.3(a)), nous partons des zones identifiées à l'étape précédente. Pour pouvoir réintégrer ces zones à l'étape 5 nous avons besoin qu'un certain nombre d'éléments restent identiques après le remaillage de la zone. Donc, nous ajoutons des éléments qui ne seront pas remaillés sur les zones frontières entre chaque partie à remailler et le reste du maillage distribué. Nous appelons ces éléments la peau. Enfin, comme le maillage est distribué, les zones peuvent être réparties sur différents processus. Il faut donc les centraliser, et les placer sur les processeurs de manière à ce qu'ils aient une charge de calcul équivalente.

L'étape 4 est le remaillage des zones par le remailleur choisi. Nous ne faisons que l'utiliser, et le considérons donc comme une boite noire. Cependant, les remailleurs ont leurs propres représentations des maillages qui sont différentes de celles utilisées par PaMPA. Il faut donc faire une conversion des structures de données de PaMPA vers celles du remailleur avant le remaillage, puis reconstruire un maillage PaMPA après. De plus, certains remailleurs comme Mmg3d font de la préservation d'éléments, c'est à dire qu'il peut ne pas remailler des éléments qu'on lui indique. Dans notre cas, la peau. D'autres remailleurs, comme Gmsh et TetGen ne font que de la préservation de face. Il faut donc extraire la peau avant de remailler (figure 1.3(b)), et la



(a) Extraction des zones à remailler

(b) Extraction de la peau d'un maillage

(c) Réintégration de la peau d'un maillage

(d) Réintégration des zones remaillées

FIGURE 1.3 – Étape de l'intégration d'un remailleur dans PaMPA

réintégrer après (figure 1.3(c)). Nous verrons plus en détails la notion de préservation d'éléments ou de faces en section 2.1.

Enfin, l'étape 5 consiste à réintégrer les parties qui ont été remaillées. Étant donné que la peau des zones n'a pas été modifiée, les nœuds des mailles n'ont pas bougé. On peut donc trouver la correspondance entre les éléments de peau, et les éléments dans le maillage distribué grâce aux coordonnées des nœuds des mailles. Une fois que l'on connaît l'emplacement de chaque partie dans le maillage distribué, on les distribue en conséquence (figure 1.3(d)).

Représentation des maillages de PaMPA

Un maillage est une représentation discrète d'un milieu physique qui peut être décrie par un graphe. Le choix fait dans PaMPA pour le manipuler est d'utiliser des listes d'adjacences, grâce à l'utilisation de deux tableaux. Le premier pour sauvegarder la liste d'adjacence des sommets, le second pour se repérer dans le premier. Prenons la figure 1.4 en exemple.

Le premier tableau est celui qui stocke les indices de la table d'adjacence, et le second est la table d'adjacence, appelons-les respectivement *ind* et *adj*. Les indices de *ind* correspondent aux numéros des sommets du graphe, et les valeurs sont les indices où commence les adjacences des sommets dans *adj*. Ceci implique d'avoir une case de plus que le nombre de sommets, afin de connaître la taille de *adj* et le dernier voisin du

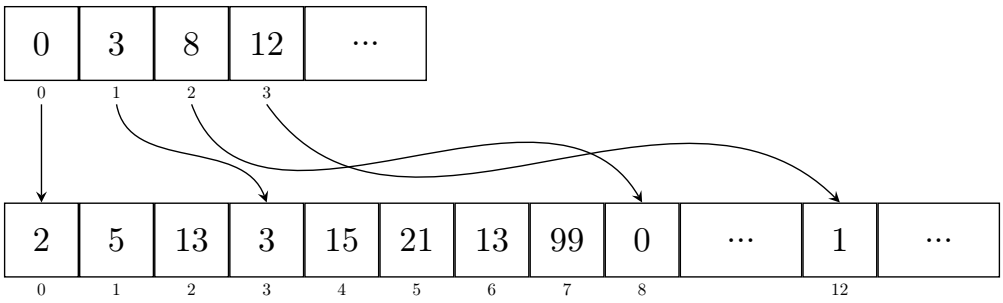


FIGURE 1.4 – Exemple de représentation csr

dernier sommet. Dans l'exemple, l'adjacence du sommet 0 commence à la case d'indice 0 de *adj*, l'adjacence du sommet 1 commence à la case 3... Les adjacences doivent être symétriques, c'est à dire que lorsqu'un sommet a a dans son adjacence un sommet b, le sommet b doit avoir le sommet a dans son adjacence. Dans la figure 1.4 nous pouvons voir que le sommet 0 à dans son adjacence le sommet 2 et que réciproquement le sommet 2 à le sommet 0 dans son adjacence. Cette symétrie nous est imposée par le fait que nous manipulons des graphes non orientés.

Ce stockage à plusieurs avantages. Le premier est qu'il ne prend que peu de mémoire. Ensuite il permet de connaître rapidement le nombre de relations total et le nombre de voisins de chaque sommet. En effet, pour avoir le degrés[6] d'un sommet n il suffit de soustraire la valeur de n dans *ind* à la valeur de la case $n+1$ de *ind*. Dans notre exemple, l'adjacence du sommet 0 commence à la case 0, et celle du sommet 1 commence à la case 3. Le sommet 0 a donc 3 voisins $(3-0)$. Tandis que le sommet 2 est de degrés 4.

Il y a un troisième tableau important. Le maillage étant composé de différent type d'entités, comme des tétraèdres, des faces, des nœuds... chacune d'entres elles se voient attribuer un numéro d'entité. La correspondance entre les sommets et les entités est sauvegardée dans ce troisième tableau.

D'autres informations peuvent être importantes et nécessiter d'être conservées tout au long du processus itératif, comme par exemple les coordonnées de chaque nœud. Ainsi, chaque information nécessitant d'être préservée est stockée dans un tableau. Pour ne pas gaspiller de l'espace mémoire, la taille de ces tableaux est égale au nombre de sommets du type d'entité à laquelle correspond l'information. Par exemple, si un maillage comprend n nœuds, le tableau contenant leurs coordonnées comprendra n cases. Par contre, cela implique d'avoir une numérotation spécifique à chaque type d'entité.

De plus, la numérotation globale du maillage distribué est différente de la numérotation d'une zone extraite et centralisée. La numérotation global doit être conservée pour tous les sommets de peau, afin de permettre la réintégration, à l'étape 5 du processus itératif, de la partie remaillée.

1.2.2 Le projet Gmsh

Né du constat qu'aucun logiciel libre n'intégrait une suite complète d'outils industriels, Christophe GEUZAINE et Jean-François REMACLE ont conçu Gmsh [3] pour combler ce manque. Il est composé de quatre modules articulés autour d'un noyau léger. Ceux-ci correspondent aux quatre étapes d'un processus industriel. En effet, Gmsh comprend un moteur CAO qui permet de créer la géométrie désirée, suit un mailleur qui permet de discrétiser cette géométrie. Vient ensuite un solveur qui n'est autre que l'outil de simulation numérique. Enfin, il y a un outil de post-traitement. Cette architecture permet à Gmsh d'être à la fois léger et rapide. De plus, grâce à son interface graphique et son propre langage de scripts, Gmsh est orienté utilisateur. Pour développer ce projet, Christophe GEUZAINE et Jean-François REMACLE ont opté pour le C++ pour ces performances et pour son utilisation du paradigme objet qui leurs ont permis de créer leur propre représentation de maillage.

Dans ce rapport nous nous intéresserons exclusivement au module de maillage/remaillage. Celui-ci comprend sa propre implantation de l'algorithme de Delaunay [1], ainsi qu'un interfaçage avec les mailleurs et remailleurs TetGen, NETGEN et Mmg3d.

6. Dans la théorie des graphes, le degrés d'un nœud correspond au nombre de voisins de celui-ci.

Afin de coupler Gmsh avec PaMPA nous devons comprendre sa représentation des maillages. Celle-ci est à deux niveaux, le premier décrit la géométrie de la CAO donnée en entrée, tandis que le deuxième correspond aux entités du maillage. La figure 1.5 montre la structure de cette représentation.

La géométrie globale est contenue dans une instance de la classe `GModel`, composée d'au moins une `GRegion`. Ces régions correspondent aux différentes parties de la CAO, et elles sont délimitées par des faces (`GFace`). Celles-ci peuvent être communes à plusieurs régions, et donc servir de frontières. De manière symétrique aux régions, les faces sont bornées par des lignes (`GEdge`), qui elles-mêmes sont définies par maximum deux points, des `GVertex`. Du plus chaque élément de dimension n connaît la liste des éléments de dimension $n+1$ qu'il délimite.

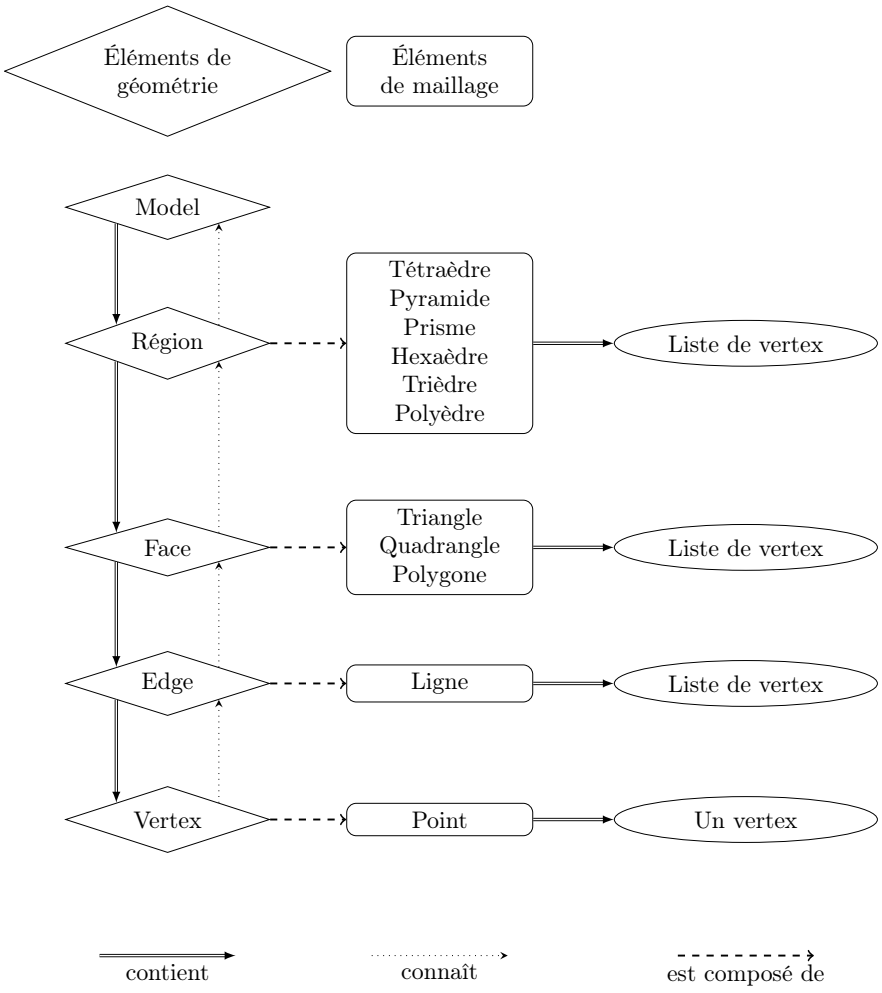


FIGURE 1.5 – Représentation de maillages à deux niveaux de Gmsh

Les différents éléments de géométrie sont constitué d'éléments de maillage de même dimension. Ainsi, les régions contiennent des tétraèdres (`MTetrahedron`), des prismes (`MPrisme`), des pyramides (`MPyramide`)... Les faces sont remplies des triangles (`MTriangle`), quadrangles (`MQuadrangle`)... Les lignes, quant à elles, contiennent uniquement des `MLigne`. Enfin, les `GVertex` ne peuvent recevoir qu'un `MPoint`. La figure 1.6 montre la correspondance entre la géométrie et le maillage dans Gmsh.

1.2.3 Problématique - Intégration

Comme nous l'avons vu, PaMPA est un outil de distribution de maillages pour le remaillage parallèle, et il a été initialement couplé avec Mmg3d. Le but de ce stage est donc de lier les projets PaMPA et Gmsh.

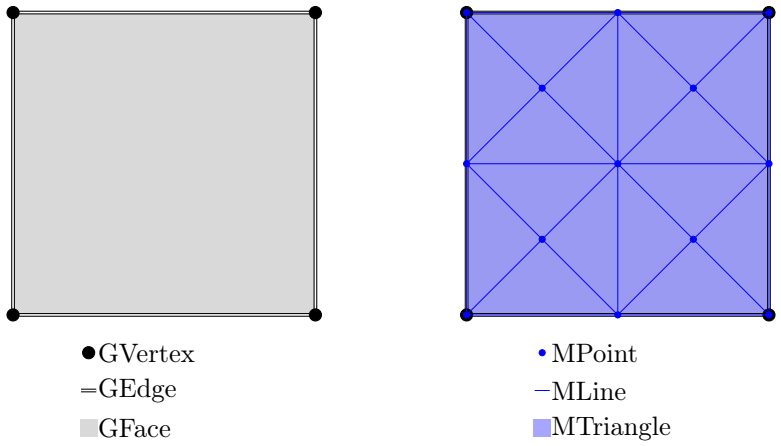


FIGURE 1.6 – Exemple de maillage Gmsh en 2 dimension

Pour cela deux axes ont été définis. Le premier est d'intégrer Gmsh comme remailleur dans PaMPA. Tandis que le second est, au sein de Gmsh, de se servir de PaMPA comme bibliothèque de distribution de maillages.

Nous commencerons par voir les étapes préliminaires ayant abouti à l'intégration de Gmsh comme remailleur au sein de PaMPA. Dans un second temps nous décrirons le travail accompli pour utiliser PaMPA comme bibliothèque de distribution de maillages au travers de Gmsh. Enfin, nous conclurons en rappelant le travail effectué et en décrivant les points que nous n'avons pu approfondir.

Chapitre 2

TetGen et Gmsh comme remailleurs dans PaMPA

Afin de tester la généricité de PaMPA nous lui intégrons le remailleur Gmsh. Nous allons voir les étapes nécessaires à son intégration, ainsi que le travail préliminaire qui a été d'intégrer le remailleur TetGen. Puis nous expliquerons le protocole de tests mis en place.

2.1 Mise en œuvre

La figure 2.1 rappelle les cinq étapes du processus itératif de PaMPA. Comme nous pouvons le voir le remaillage se situe à l'étape 4. La figure nous montre aussi qu'il y a des étapes intermédiaires entre les étapes 3 et 4 et entre les étapes 4 et 5. Reprenons donc ces étapes.

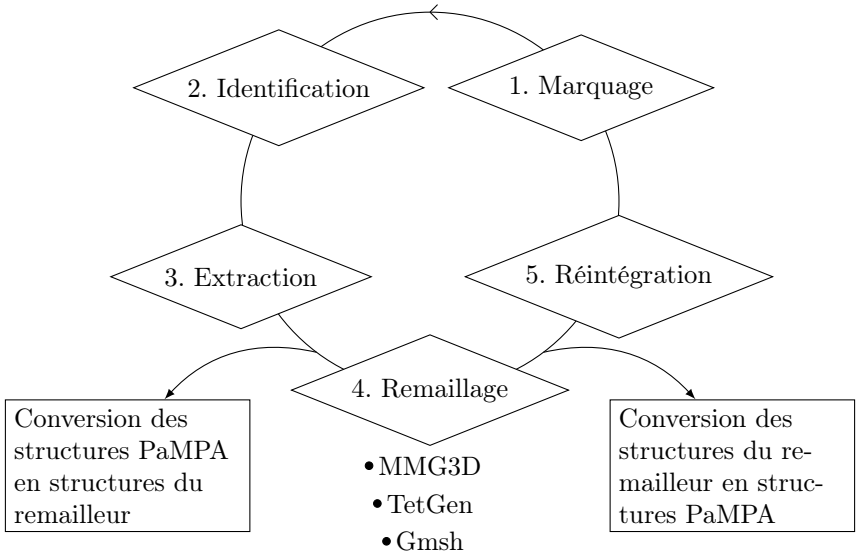


FIGURE 2.1 – Remailleurs intégrés dans PaMPA

L'étape 3 consiste à extraire du maillage global distribué une zone identifiée et à la centraliser. En vue de la réintégrer à l'étape 5, des éléments sont ajoutés sur la frontière entre la partie extraite et le reste du maillage. La peau peut servir à la réintégration, car Mmg3d, le remailleur initialement intégré à PaMPA, permet la préservation d'éléments. C'est à dire que nous pouvons marquer des éléments à ne pas toucher, et Mmg3d ne les modifiera pas. Cependant, contrairement à Mmg3d, TetGen et Gmsh ne font que de la préservation

de faces, et non de la préservation d'éléments. La figure 2.2 montre un maillage donné à un remailleur qui préserve les éléments, et le même maillage donné à un remailleur qui ne fait que de la préservation de faces.

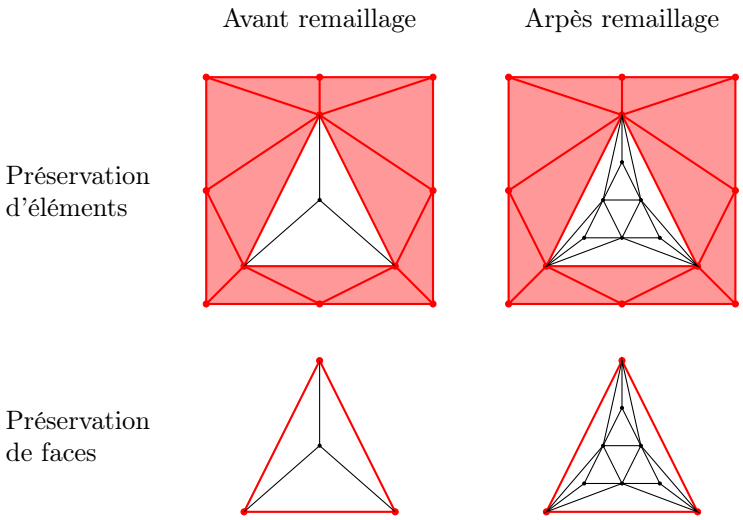


FIGURE 2.2 – Exemple de préservation d'éléments et de faces

Afin de pouvoir utiliser la peau lors de l'étape 5, nous devons séparer la peau et la zone à remailler. Pour cela il faut identifier les éléments (nœuds, faces, volumes) qui appartiennent à la peau, ou à la partie interne, et ceux qui sont à la frontière et qui appartiennent au deux. Un fois cela fait, nous pouvons reconstruire deux maillages PaMPA.

Pour l'étape 4, il faut convertir les structures de données de PaMPA en structures de données du remailleur choisi. Dans le cas où le remailleur supporte la préservation d'éléments, nous convertissons le maillage issu de l'étape 3. Tandis que dans le cas où le remailleur ne fait que de la préservation de faces, nous ne convertissons que la zone à remailler, et nous laissons la peau inchangée. Après la conversion des structures, le remailleur choisi remaille séquentiellement la zone identifiée à l'étape 2.

Une fois le remaillage fini, nous convertissons le maillage obtenu (en structures de données du remailleur) en maillage PaMPA. Si le remailleur permet la préservation d'éléments nous réintégrons directement le nouveau maillage centralisé dans le maillage global distribué. Sinon, nous fusionnons le maillage obtenu avec la peau.

Pour ajouter le maillage de peau au maillage issu de l'étape 4, nous comparons les coordonnées des nœuds. La préservation des faces ne modifiant aucun élément sur le tour du maillage nous pouvons trouver les nœuds du maillage interne correspondant aux nœuds de la peau. Une fois cette correspondance trouvée, nous fusionnons les listes d'adjacences des éléments situés sur la frontière, pour ensuite recréer un maillage PaMPA complet.

Nous nous sommes heurtés à un problème avec TetGen car il rajoute des sommets de types face sur le bord du maillage (sans modifier les nœuds). Étant sur la frontière entre le maillage interne et le maillage de peau, mais n'ayant pas de correspondance dans le maillage de peau, nous avons dû retirer ces faces, en modifiant les listes d'adjacences en conséquence.

La conversion entre les structures de données de PaMPA et de TetGen avait déjà été commencé avant mon arrivée. Il a donc fallu la finir, ainsi que faire la conversion entre les structures de données de PaMPA et de Gmsh.

2.2 Protocole de tests

La conversion entre les maillages PaMPA et TetGen étant commencée à mon arrivée, je l'ai complété pour valider l'intégration de TetGen en séquentiel. Suite à quoi nous avons testé le module d'extraction de la peau, pour tester TetGen en parallèle. La même méthodologie a été appliquée avec Gmsh. Nous commençons par valider la conversion de maillage en séquentiel, avant de la tester sur un maillage distribué.

Chapitre 3

Gmsh utilise PaMPA

Nous venons de voir l'intérêt de PaMPA dans le cas où le maillage évolue au cours de la simulation. En effet, PaMPA permet de remailler efficacement des maillages imposants. Ceci implique que le maillage initial est de taille plus modeste, et a été maillé de manière séquentielle. Cependant, même les simulations ne nécessitant pas de remaillage, peuvent avoir besoin de maillages fins, pour avoir une solution précise. Ces maillages sont donc eux aussi de plus en plus gros, et il devient critique de les mailler séquentiellement. Nous avons donc pour objectif d'utiliser PaMPA pour mailler en parallèle une géométrie à travers Gmsh, comme l'illustre la figure 3.1.

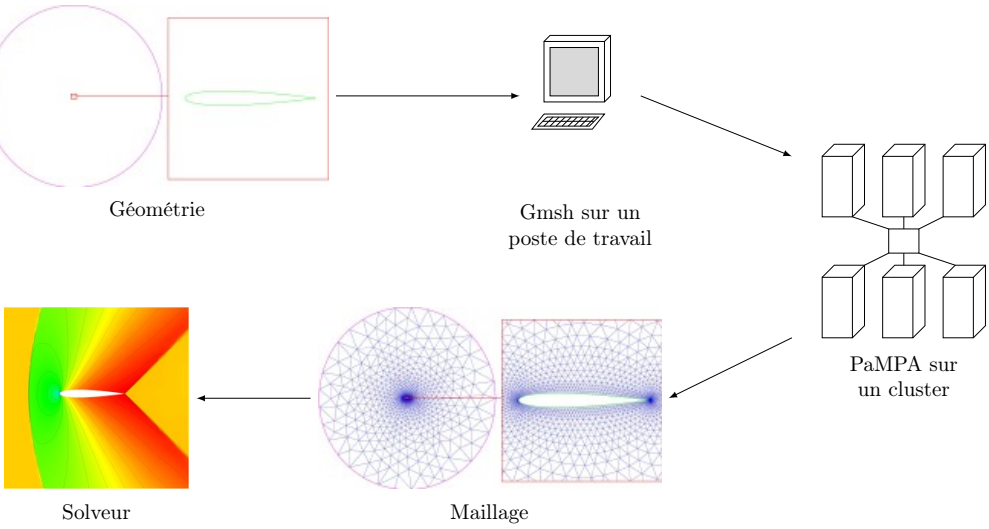


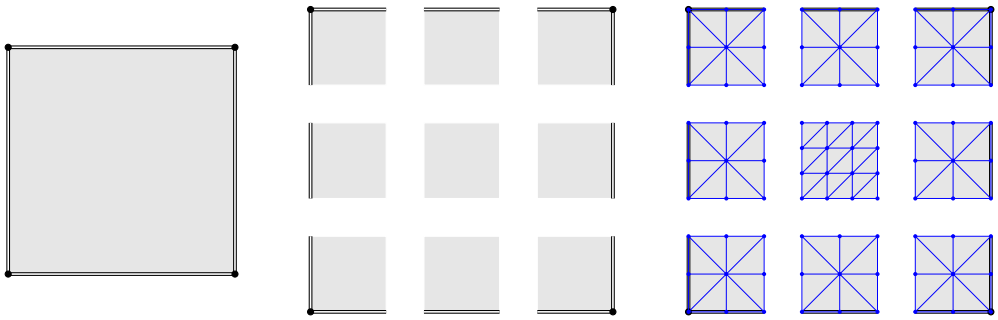
FIGURE 3.1 – Gmsh utilisant PaMPA sur un cluster

Jusqu'à présent PaMPA permet uniquement de remailler sur un cluster, notre but ici est de détourner PaMPA pour mailler directement une géométrie sur un cluster. Pour cela il est nécessaire de découper cette géométrie en parties qui auront une charge de calcul équivalente, afin d'utiliser au maximum tous les nœuds de calcul, et donc d'avoir une utilisation optimale du cluster.

Ce découpage pose deux problèmes. Le premier est de déterminer les endroits où découper la géométrie afin d'avoir un maillage exploitable. En effet, il peut y avoir des conditions aux limites d'objets de la géométrie. Et le mailleur peut mailler différemment l'interface entre deux objets géométriques si elle est sur un bord du maillage, ou au milieu. Il faut donc veiller à couper la géométrie de manière à respecter ces conditions aux limites.

Le second problème concerne l'assemblage des parties qui ont été maillées, car il peut y avoir des zones qui se trouvent au milieu d'un objet homogène. C'est à dire que ces zones n'ont ni faces, ni arêtes ni nœuds de la géométrie pour se repérer. Ceci implique que deux parties voisines peuvent avoir des nœuds différents

sur leurs frontières communes, ce qui rend impossible la réunification des parties, ou cela pourrait générer des maillages non conformes (ex : des points au milieu d'une arête).



Géométrie initiale | Géométrie distribué sur neuf processus | Maillage distribué incorrecte

FIGURE 3.2 – Géométrie distribuée, puis maillée

Sur la figure 3.2, qui présente ce problème, nous pouvons constater que les huit sous-parties extérieures du maillage peuvent être ré-assemblées aisément, car toutes les frontières sont symétriques. En revanche la partie centrale ne peut pas être rattachée au reste, puisqu'elle a quatre nœuds sur chaque côté, alors que ses voisins n'en ont que trois. Il n'est donc pas possible de réassembler ce maillage.

Dans un premier temps, afin de nous soustraire de ces problèmes contraignants, nous allons utiliser Gmsh pour générer un maillage grossier, qu'il transférera à PaMPA pour qu'il le remaille en parallèle.

3.1 Principe

À partir d'une géométrie donnée, Gmsh crée un maillage initial assez grossier de manière séquentielle. Puis ce maillage est converti en maillage PaMPA afin qu'il soit remaillé parallèlement. Enfin, ce maillage est reconverti en maillage Gmsh distribué.

Chapitre 4

Conclusion

Nous venons de voir que pour étudier des phénomènes physiques l'industrie comme la recherche ont de plus en plus recours à la simulation numérique. Celle-ci s'appuie sur des maillages, qui sont des représentations discrètes du milieu physique étudié. Ils permettent de calculer des valeurs physiques en des points précis. La précision des simulations dépend directement de la taille et de la qualité des éléments de maillage. Les numériciens ayant toujours des besoins en précisions grandissant, les maillages deviennent de plus en plus gros. Ce qui rend leur génération ou leur remaillage compliqué sur un poste de travail.

Nous avons donc étudié une bibliothèque pour paralléliser le remaillage de maillages imposants, qui se nomme PaMPA. Cet outil est basé sur la distribution de maillages et sur un processus itératif en cinq étapes, qui permet de remailler séquentiellement le maillage partie par partie. Nous avons continué en analysant le remailleur séquentiel Gmsh.

Afin de tester la généricité de la bibliothèque PaMPA, nous lui avons intégré les remailleurs séquentiels TetGen et Gmsh. Nous avons détaillé les étapes qui nous ont permis de mener à bien ce projet, et nous avons expliqué les difficultés que nous avons rencontré.

Enfin, nous avons exposé une utilisation de PaMPA qui permet de générer des maillages importants en parallèle. Ceci en se basant sur Gmsh qui fait appel à PaMPA pour pouvoir utiliser la puissance de calcul d'un cluster.

Il nous reste cependant à identifier quelques bogues qui ne nous ont pas permis de présenter des résultats concrets. De plus, le temps nous a manqué pour trouver une méthode de distribution de géométrie, afin de générer le maillage souhaité sans étapes intermédiaires. Le temps restant sera consacré à ces tâches.

Bibliographie

[1] Triangulation de deulaunay. `https://fr.wikipedia.org/wiki/Triangulation_de_Delaunay`. Dernière modifications le 26 avril 2016.

[2] Cédric Lachat : *Conception et validation d'algorithmes de remaillage parallèles à mémoire distribuée basés sur un remailleur séquentiel*. Thèse de doctorat, Université Nice Sophia Antipolis, 2013.

[3] Christophe Geuzaine et Jean-François Remacle : Gmsh : a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING*, 2009.